

A Method for Extracting Symmetry from Meshes

Simge Tekin
METU, Turkey

Yusuf Sahillioğlu
METU, Turkey

Abstract

Motivated by the importance and universality of symmetry detection problem, we present a simple yet effective solution that is able to identify symmetric correspondences as well as the symmetry axis curve on a given mesh model embedded in 3D. We consider global intrinsic reflective symmetry that is invariant to isometric, i.e., distance-preserving, pose changes of the input model. The key insight to our solution is to minimize the deviation from isometry in the constructed symmetric self-maps by generating base vertices and feature vectors in coarse-to-fine manner. Comparison with the state-of-the-art methods reveals the timing and simplicity advantages of the proposed method, whereas our accuracy is almost tied with them. Our method does not require any initialization and does not demand restrictions on topology, tessellation, and resolution.

1. Introduction

Symmetry does not only serve aesthetic considerations, e.g., beauty of human faces, but also physical and algorithmic optimality principles. We can exemplify the physical aspect with architectural constructions such as LEGOs [Xu et al. 2019], and the latter with geometry processing algorithms such as automatic discovery of surface maps [Liu et al. 2012; Sahillioğlu 2020], surface segmentation [Dessein et al. 2017], and surface embedding [Yoshiyasu et al. 2016].

We address the important problem of symmetry detection in geometry processing. Namely, we detect reflectively symmetric correspondences as well as the reflective symmetry axis on digital models embedded in 3D space. Our output correspondences and axis are invariant to the isometric, i.e., distance-preserving, pose changes of the query model. We, consequently, consider reflective intrinsic symmetry. Compared to the well-studied and easier extrinsic counterpart, e.g., symmetry of Eiffel Tower, extraction of intrinsic symmetry information requires a huge search space involving rotations, translations and bends, e.g., human motions. We consequently cast an optimization problem that can efficiently and effectively analyze this space.

Key insight to our symmetric map computation is the preservation of pairwise geodesic distances to the carefully selected and matched base vertices. Symmetry axis extraction is performed afterwards via voting once the symmetric matches and side information of the vertices are made available. Our method is resilient to small deviations from isometry that are due to imperfections of the modeling process and/or geometry discretization errors. It handles the extrinsic symmetry naturally, yet replacing intrinsic geodesic distances with Euclidean distances will lead to a faster extrinsic method in this simpler scenario. Our method, however, does not handle rotational symmetries and does not support symmetry extraction on partial shapes where some of the symmetric regions are missing, e.g., human with no left leg.

2. Related Work

For the reflectional symmetry, which is the focus of our paper, papers work on extrinsic and intrinsic levels where the former and latter consider the Euclidean and geodesic distance compatibilities, respectively [Mitra et al. 2013]. Our algorithm operates on intrinsic level, hence the focus of this literature review.

Most of the objects around us do not lose their reflective symmetry even if they are isometrically bent, e.g., eyeglasses in and out of pocket, humans sitting and standing, etc. Despite being this common, the first study dealing with the symmetry of bendable/articulated isometric objects is seen only recently in 2007 [Raviv et al. 2007]. It adapts generalized multidimensional scaling, a method that embeds one surface onto the other for surface pair matching, in order to detect global reflective intrinsic symmetries. Similarly, [Kim et al. 2010] converts a matching method that works on surface pairs into their method that maps surface to itself. Specifically, it votes through the Möbius transformations defined over genus zero surfaces. Another voting-based algorithm operates on a scalar field to extract partial symmetries [Xu et al. 2009]. The probabilistic framework in [Xu et al. 2012], on the other hand, detects intrinsic symmetry by estimating the likelihood two point pairs belong to symmetries at the same scale. From a small number of symmetric pairs of points, [Wang and Huang 2017] determines intrinsic symmetries that are represented as linear transformations of functional space on shapes.

Symmetry computation under mild noise is also explored. To this end, [Ovsjanikov et al. 2008; Nagar and Raman 2018] use the heat diffusion properties via the eigenfunctions of the Laplace-Beltrami operator in order to produce self-maps, i.e., mapping the right side of the surface to its own left. The same spectrum is employed later in [Raviv et al. 2010b] in an effort to extend [Raviv et al. 2010a] into a more stable method under a certain amount of topological noise. Functional maps defined over this spectrum are also utilized for intrinsic reflective symmetry detection [Liu et al. 2015; Ren et al. 2020]. Alternatively, skeletons can be used for intrinsic sym-

metry detection in order to alleviate the problems due to the distorted geodesics under such noise [Jiang et al. 2013; Wang et al. 2019].

A different branch of methods explicitly change the spatial coordinates of the input model in order to make it look more symmetric [Mitra et al. 2007; Sahillioğlu and Kavan 2016]. Direct Euclidean distances or other extrinsic tactics such as smooth shape diameter signature [Graus et al. 2021] and fully-convolutional symmetry prediction network [Shi et al. 2020] can then be utilized on these new forms.

3. Method

Given a source and target model, our method operates in three steps. At first, we extract symmetric correspondences of the extreme points and divide the model into five regions. In the second step, based on the symmetric correspondences, we map the points in source regions to their intrinsically symmetric target region correspondences. We finally extract the symmetry axis curve based on these mappings. Note that although our method can be applied to any model with bilateral symmetry (Section 5), we use human hand and human foot metaphors in order to make the reading in the sequel easier.

3.1. Extreme Points Mapping

We find correspondences of extreme points, namely, tip of hands and feet. Upcoming denser maps will be based on these extreme points matches. We observe that the embeddings of models in spectral domain concretizes the limbs, making the extreme points farther from each other. We embed the meshes in spectral domain as defined in [De Silva and Tenenbaum 2002]. To this end, we construct the affinity matrix by using evenly-spaced hundred landmark vertices [Eldar et al. 1997], and obtain embeddings of them at first via classical multidimensional scaling as described in [Sahillioğlu and Yemez 2012]. Then we embed the remaining vertices on the mesh by scaling the geodesic distances to these landmarks. In this way, we did not need to compute the geodesics for all vertices, saving both computation and storage.

Afterwards, we sample four vertices that are geodesically farthest from each other [Eldar et al. 1997]. These vertices would correspond to the tip of hands and feet, forming the extreme points set $E = \{h_1, h_2, f_1, f_2\}$.

In order to detect hand-to-hand and foot-to-foot correspondences, we follow a number of steps, some of them having side effects that will be useful later. Firstly, we approximate the average geodesic distance function (AGD) [Kim et al. 2010] by approximating geodesic distances on surface using Dijkstra’s algorithm, and smoothing AGD over 1-ring neighborhoods. To speed up the computation of AGD, we use hundred vertices on the embedded mesh model that are again determined by farthest point sampling algorithm [Eldar et al. 1997]. Note that we did not use the hundred

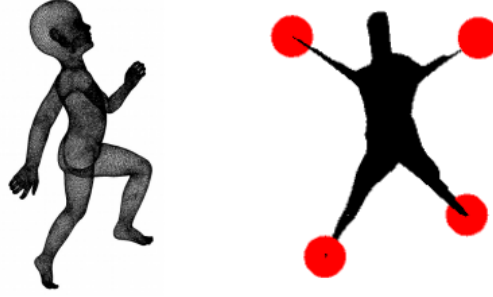


Figure 1. Kid model (left) and its 3D spectral embedding with extreme points (red).

sample vertices that we used to construct affinity matrix since the geodesics computed for them was for the mesh in spatial domain. Our experiments suggest that geodesics on spectrally embedded mesh yield better AGD values for the upcoming S_1 and S_2 construction. We form the set S_1 with the local extrema (namely maxima) of AGD values over 1-rings. For the purpose of computing minimal geodesic distance (MGD) that is assigned to every vertex of the mesh, we use S_1 as described in [Kim et al. 2010]. Namely, MGD for a given vertex is the geodesic distance to the closest vertex in the set S_1 . Similarly, local maxima of the MGD values form the set S_2 . Figure 2 shows S_1 and S_2 on a model. These sets will be used later in Section 3.3.

We then sort the four extreme vertices with respect to their MGD values and group the two with the smallest MGDs and the other two with the largest MGDs. The vertices in these groups are naturally the symmetric correspondences of each other. For low-resolution models with less stable Dijkstra-based geodesics, performing this operation with AGD values instead of MGD values is more successful while extracting hand-to-hand and foot-to-foot correspondence. The main reason is the smoothness of AGD which is defined by integrating geodesic distances. MGD, on the other hand, is defined using single geodesic distances and therefore it tends to be irregular and unstable. Thus, we use AGD while matching extreme vertices on low-resolution meshes, e.g., less than 30K vertices, and MGD for the high-resolution models.

3.2. Initial Region Assignments

Next step is to detect the two extreme vertices that are on the same side of the symmetry axis curve, e.g., left hand and left foot. In order to achieve this, we use geodesic distances between the extreme vertices that are not symmetric correspondences of each other, i.e., one extreme vertex from one group, e.g., $\{h_1, h_2\}$, and the other extreme from the other group, e.g., $\{f_1, f_2\}$. If $d_g(h_1, f_1) < d_g(h_1, f_2)$, then h_1 and

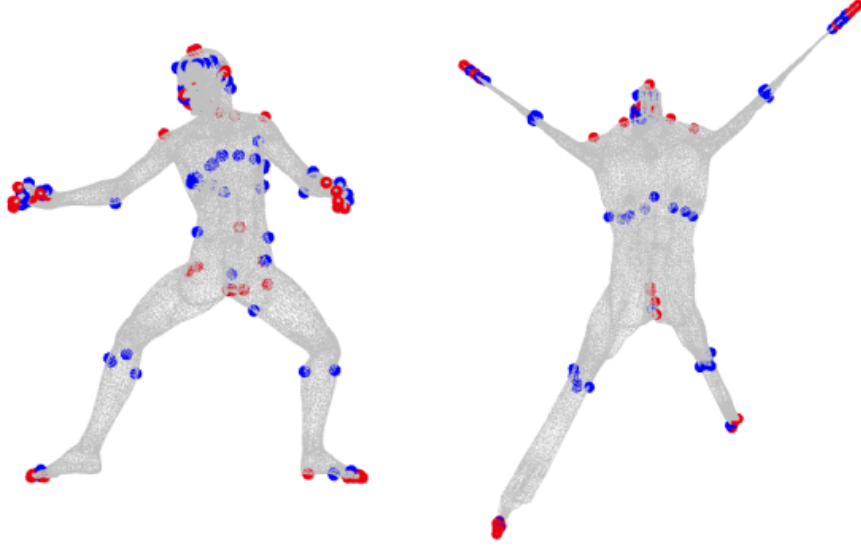


Figure 2. Sets S_1 (red) and S_2 (blue) on David model in spatial domain (left) and after spectral embedding (right).

f_1 are assumed to be on same side of the symmetry axis curve as they would be in a standard human body. After this step, we revert the model back to spatial domain, by changing spectral coordinates of vertices with the values they have in the original form.

Using midpoints of the extreme vertex correspondences, at first we define two source regions and two target regions on mesh model M :

$$R_{\text{source1}} = \{v \in M | d_g(v, h_1) < d_g(v, m_{h_1, h_2})\} \quad (1)$$

$$R_{\text{target1}} = \{v \in M | d_g(v, h_2) < d_g(v, m_{h_1, h_2})\} \quad (2)$$

$$R_{\text{source2}} = \{v \in M | d_g(v, f_1) < d_g(v, m_{f_1, f_2})\} \quad (3)$$

$$R_{\text{target2}} = \{v \in M | d_g(v, f_2) < d_g(v, m_{f_1, f_2})\} \quad (4)$$

where m_{h_1, h_2} is the midpoint of the extreme match $h_1 \leftrightarrow h_2$ and m_{f_1, f_2} is defined similarly. Now, we have two source-target region pairs that are symmetrically equivalent, e.g., left arm with shoulder and right arm with shoulder. We also know that two sources are on the same side with respect to the symmetry axis curve while two target regions are on the other side. The remaining vertices that are not in these four regions form a fifth region that works as both source and target, hence no sixth region is required. As a result of this process, we obtain four regions for which we know

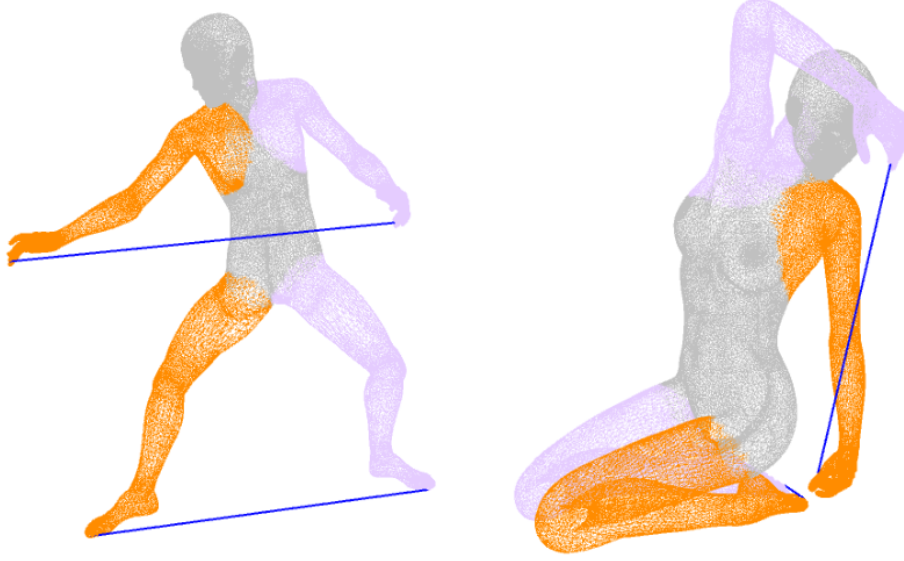


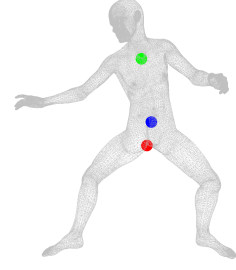
Figure 3. Source regions (orange), target regions (purple), and the fifth region (gray) along with the mapping of extreme vertices. Regions including an extreme vertex and its correspondence has source-target relationships.

pairwise symmetrical source and target relationship and one region without that information (see Figure 3). Purpose of extracting these regions is to narrow down the search space to a target region instead of whole mesh while processing a vertex in source region.

3.3. Dense Symmetric Mapping and Symmetry Axis Curve

The key observation that drives our dense symmetric mapping computation and symmetry axis computation is the following. If some vertex $p \in M$ and $q \in M$ are symmetrical correspondences of each other, and some $s \in M$ is determined to be on symmetry axis curve, i.e., midpoint of the path connecting p and q , then $d_g(p, s)$ and $d_g(q, s)$ should be equal independent of the pose. Also, if p' and q' are symmetrical correspondences, $d_g(p, p')$ and $d_g(q, q')$ are expected to be equal. In practice, however, since bends caused by pose can distort shortest paths, $d_g(p, s)$ and $d_g(q, s)$ might not be equal. Our goal is therefore to use geodesic distances as a similarity measure in correspondence extraction while *minimizing* this distortion by using a group of base vertices. Consequently, the procedure is based on feature vectors that are made up of geodesic distances to some predetermined base vertices. We implement an iterative procedure where previously extracted correspondences enhance the upcoming extractions by introducing new base vertices. We proceed in 4 stages.

Stage 1: For the first stage of the dense map computation, the base vertices are determined to be i) extreme vertices, ii) m_h : midpoint of h_1 - h_2 path, iii) m_f : midpoint of f_1 - f_2 path, and iv) m_{hf} : midpoint of the path between the midpoints of h_1 - f_1 and h_2 - f_2 paths. Midpoints are shown in the inset where blue is the m_{hf} . Although the base point set is defined as $B = \{h_1, h_2, f_1, f_2, m_h, m_f, m_{hf}\}$, for feature vector construction of a vertex, we use extreme vertices that are detected to be on the same side as the vertex (midpoints are always used regardless of the side). For example, if h_1 , f_1 and source regions are on the same side, then for a vertex in one of the source regions, base vertices that are used to construct the feature vector are $\{h_1, f_1, m_h, m_f, m_{hf}\}$. Similarly, for a vertex in one of the target regions, these vertices would be $\{h_2, f_2, m_h, m_f, m_{hf}\}$.



In the first stage, we perform the mapping on S_1 that we formed before (Section 3.1). Reason for this is that S_1 is a symmetry invariant vertex set, and includes much smaller number of elements compared to the whole mesh. This gives us a small area to search for robust mappings. We divide the vertices in S_1 into source and target regions according to the region they fall into. We ignore the vertices on S_1 that are on the fifth region on this stage. Thus, only the vertices that are on both S_1 and one of the arm and leg regions presented on Figure 3 are matched.

Feature Vector Creation: The fundamental component that is required for mapping at any stage is the feature vector, which we attach to all vertices. For a vertex, v , on the same side as h_1 and f_1 , e.g., source region, the feature vector is created initially as a 5 dimensional vector ρ :

$$\rho = (d_g(v, h_1), d_g(v, f_1), d_g(v, m_h), d_g(v, m_f), d_g(v, m_{hf})) \quad (5)$$

Feature vector for a vertex v' in the corresponding target region of that source region is then the following:

$$\rho' = (d_g(v', h_2), d_g(v', f_2), d_g(v', m_h), d_g(v', m_f), d_g(v', m_{hf})) \quad (6)$$

To map vertices from source region to target region, we choose the vertex v' in the target region that minimizes D :

$$D = \sum_{i=1}^5 \left(\frac{\rho_i - \rho'_i}{\rho_i} \right)^2 \quad (7)$$

We use these feature vectors to map vertices in S_1 that fall into two source and two target regions. At the end of mapping procedure for a region, we extract midpoints of at most five correspondences with the smallest D values, i.e., if there are more than

five matches between two regions, we pick the best five. Distances to these midpoints are appended as new dimensions to the growing feature vector of the vertex v . We also append the distances to the endpoints of the S_1 -based correspondences that are on the same side as v . Similar appends take place for v' as well.

Stage 2: For the second stage, we match the set S_2 obtained in (Section 3.1). Similar to what we do in the mapping of S_1 , we divide S_2 into two source and two target regions depending on their positions. We then compute the mapping while adding new dimensions to feature vector similar to Stage 1, e.g., we append distances to the midpoints from all region-to-region mappings as well as distances to the endpoints at the correct side.

Stage 3: Thus far, each match provides two dimensions to the feature vector, namely distance to its midpoint and to the endpoint of correct side. In this stage, we remove the dimensions due to the endpoints and keep only the midpoint-based dimensions. The reason is that, the endpoints can only be selected from the restricted S_1 and S_2 sets and hence are prone to errors; errors are tolerable since S_1 and S_2 are symmetry-invariant vertices but still we observe that they are not as stable as the midpoints they provide. For example, a match running from source shoulder to target armpit has slightly incompatible endpoints but a stable midpoint that is on the ideal symmetry axis, which makes it useful as a feature vector base vertex.

For this stage, we sparsely sample hundred [Eldar et al. 1997] vertices on two source regions and the fifth center region (Fig. 3). This stage aims to find correspondences of to these samples. For all sample vertices in the source regions, we again seek the match in their corresponding target regions, where all region vertices are considered as match candidates. For all sample vertices in the fifth region, we consider each vertex from the same region as a match candidate.

We then take the best 20 best matches (minimizing D) from each region-to-region mappings and use their midpoints to grow the feature vectors further. For the feature vector of the vertex v , we also append the distances to the endpoints of these correspondences that are on the same side v .

Stage 4: We perform the mapping between all hundred samples in this stage but using the latest feature vectors that include the distances to both the endpoints and midpoints of the matches obtained in Stage 3. Note that we include the endpoints as well since they are more reliable and evenly distributed than the restricted sets S_1 and S_2 that we discard after Stage 2. We keep the best 30 matches (minimizing D) from each region-to-region mappings.

For all correspondences, we evaluate the side of the vertices that form the correspondence as described in the sequel. If side of both vertices in a given correspondence is the same, we eliminate that correspondence. Finally, in order to detect the vertices that are on the symmetry axis region, we decide the side of each vertex as either source or target. We then add the vertex to the symmetry axis curve if its 1-ring

neighborhood consists of a hybrid set of source and target side labels.

Side Decision: We note that our algorithm is not sensitive to the potential mistakes in the initial region assignments of Section 3.2. These mistakes rarely arise due to discretization errors when left hand happens to be closer to the right foot instead of the left foot; in this case the left hand region, i.e. the left arm with shoulders (Figure 3), and the right foot region, i.e., the right leg, are labeled as source regions although they are at different sides of the symmetry axis in reality. This is definitely not a problem during the symmetric correspondence computations because foot region is mapped to the other foot region regardless of what their labels are, i.e., source or target. This is, however, a problem during symmetry axis extraction which depends on the side labels of each mesh vertex. As a solution to this problem, we reevaluate the side label of the newly matched vertex and possibly change it from source to target or vice versa. The change is made for vertex v depending on its distances to the previously matched vertices whose sides are known. Specifically, we count the number of endpoints in the previous matches that are closer to v than their symmetric endpoints. We accept v to be on the side for which the count is higher. Thus, a vertex that is determined to be on left side can be determined to be on right side later.

4. Results

We conducted experiments on three different datasets: SCAPE [Anguelov et al. 2005], TOSCA [Bronstein et al. 2008], and KIDS [Rodola et al. 2014] in order to obtain quantitative and qualitative results of the extracted symmetry information by our algorithm. Symmetry information consists of the symmetric correspondences and the symmetry axis.

SCAPE, a reconstructed pose sequence of a human actor contains 71 non-uniformly sampled 12K-vertex models with the fixed template connectivity. We use the high-resolution Cat, David, Gorilla, Michael, Victoria, and Wolf objects from TOSCA, each class representing the motion of an articulated object with 11, 7, 4, 18, 12, and 3 meshes, respectively, where humans have about 50K vertices and the others about 25K. KIDS dataset, consisting of 32 meshes of two kid models, has fixed connectivity over 60K vertices. Ground-truth information for a subset of ~ 20 symmetric vertices are given for SCAPE and TOSCA by [Kim et al. 2010], whereas this information is given for all the vertices of KIDS models. Ground-truth subset corresponds to the informative parts, e.g., ears and eyes.

We employ the Correspondence Rate and Mesh Rate evaluation metrics that are used by Möbius Transformations (MT) [Kim et al. 2010] and Fast and Accurate Intrinsic Symmetry Detection (FAISD) [Nagar and Raman 2018] methods in order to provide fair comparisons. We also introduce Geodesic Error in order to gain addi-

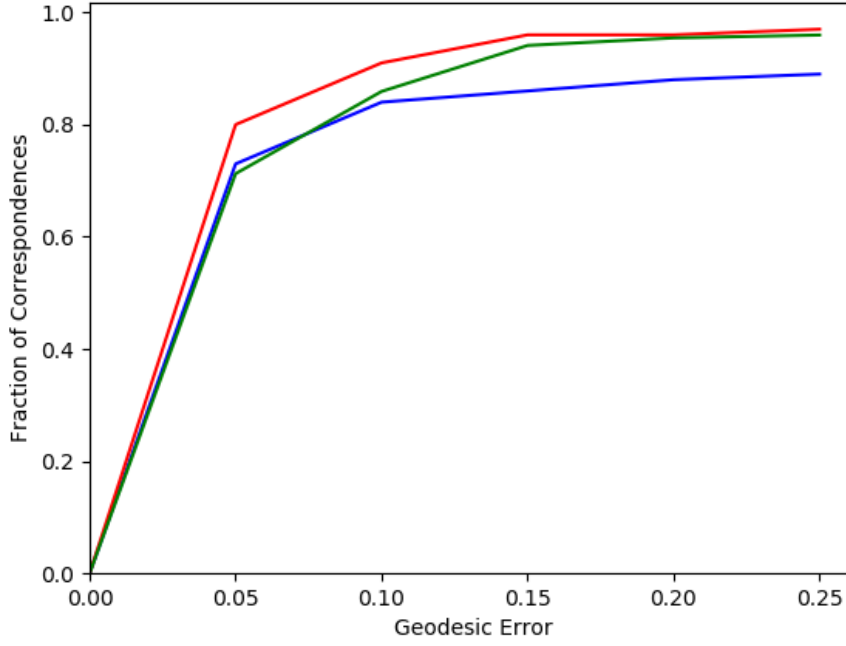


Figure 4. Geodesic Error plots of our results on KIDS (blue), SCAPE (green), and TOSCA (red) datasets.

tional insight about the accuracy of our results.

1. Geodesic Error: The fraction of correspondences that has geodesic distance between $m(v)$ and $g(v)$ less than a given amount where the maximum distance over the surface is normalized to 1. Here, $m(v)$ is the automatically computed match of v whereas $g(v)$ is the ground-truth match.

2. Correspondence Rate: The percentage of correspondences with geodesic distance between the computed and ground-truth correspondence of the vertex less than $\sqrt{\frac{\text{area}(M)}{\pi N}}$. We take $N = 20$ as used in MT.

3. Mesh Rate: The percentage of meshes for which the correspondence rate above is more than 75%, as used in MT.

Table 1 shows Correspondence Rate and Mesh Rate whereas Figure 4 depicts the average Geodesic Error of KIDS, SCAPE and TOSCA datasets. We can observe the fraction of mappings that has smaller geodesic error than 0.05, 0.1, 0.15, 0.2 and 0.25 while maximum possible error is 1. The plot and rates of KIDS dataset represent the evaluation of all correspondences extracted on Stage 4 of our algorithm (~ 150 matches), while SCAPE and TOSCA counterparts represent the evaluation of corre-

Table 1. Performance of our algorithm on SCAPE, TOSCA and KIDS datasets.

Dataset	Corr. Rate	Mesh Rate
SCAPE	81%	75%
TOSCA	87%	88%
KIDS	84%	88%

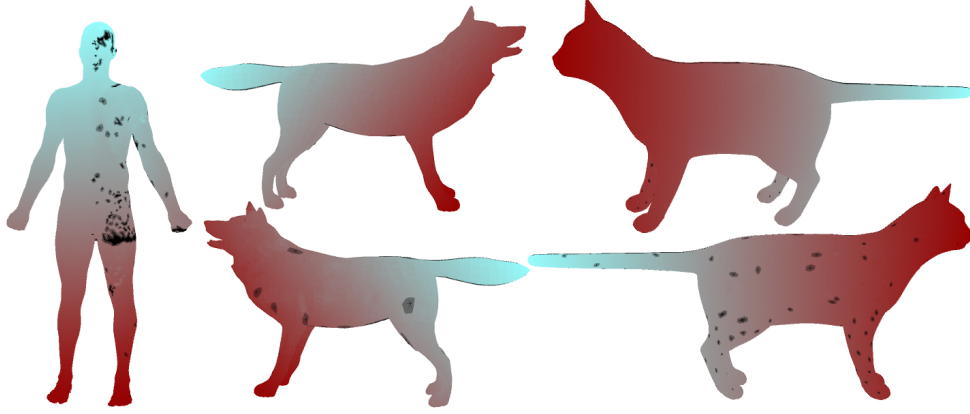


Figure 5. Transfer of colors on one side to the other using our dense symmetric maps on SCAPE human and TOSCA animals. Unmatched vertices are shown in black.

spondences for provided subset of vertices with ground-truth data (~ 20 matches). We use our Stage 4 mapping to define the base vertices and feature vectors that match any other vertex in dense correspondence mode (Figure 5). Similarly, based on our Stage 4 mapping we match the specific ~ 20 vertices which have ground-truth correspondences as designated by the MT method.

Figure 6 displays symmetry axis curves obtained by our algorithm. We observe accurate curves under large isometric deformations thanks to our robust dense vertex side labeling procedure that is based on the ~ 150 matches coming from our Stage 4. In addition to the symmetry axis curves, Figure 7 and Figure 8 demonstrate our matches for which ground-truth data is available (MT). Since the ground-truth data is available for all vertices in KIDS, we display all of our Stage 4 matches along with the extracted symmetry curve in Figure 9.

In our experiments, we observe that, especially for low resolution meshes, the Correspondence Rate for failed models is very low while for successful models this rate is high. Thus, failed models significantly decrease the average Correspondence Rate of a dataset (Tables 1-2). We compare the results of our method with MT [Kim et al. 2010] and FAISD [Nagar and Raman 2018] on SCAPE and TOSCA only as they

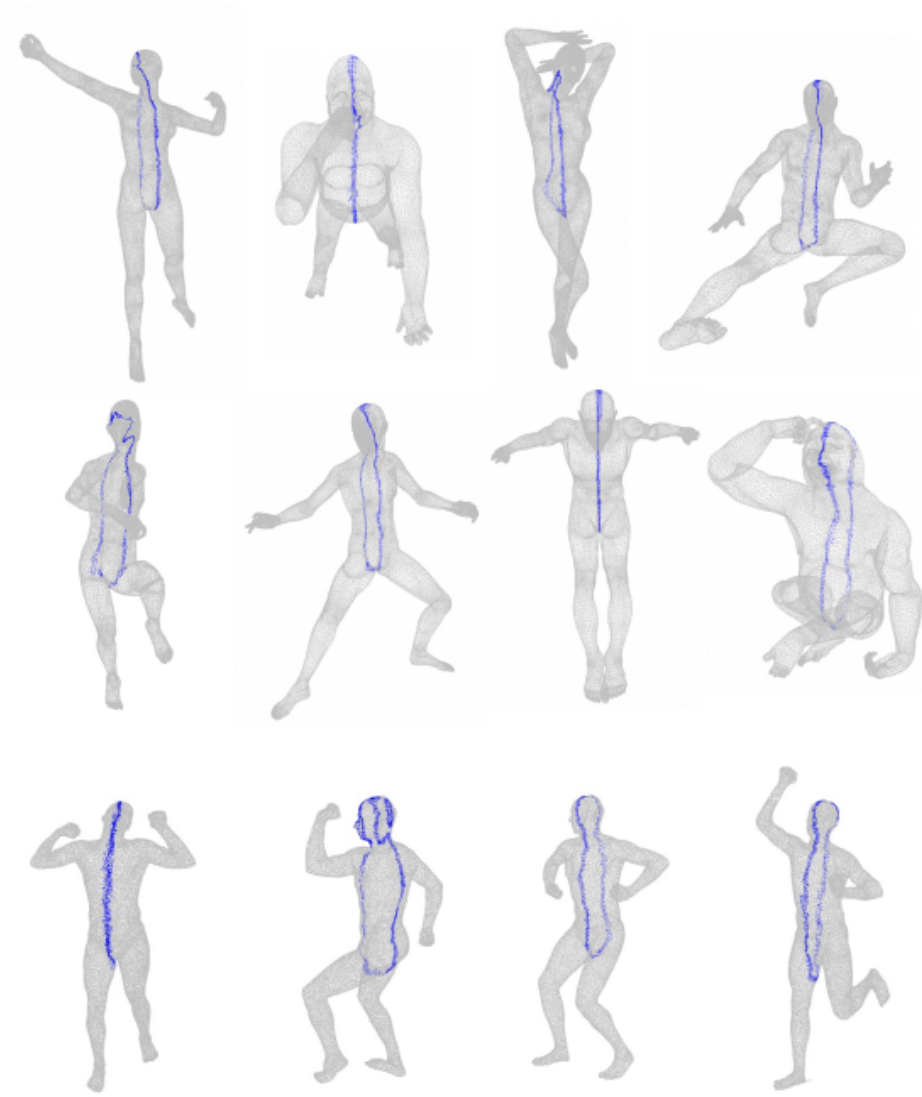


Figure 6. Global intrinsic reflective symmetry axis curves extracted by our algorithm on TOSCA (first two rows) and SCAPE (last row) models.

have not used KIDS dataset in their test suites (Table 2). We avoid the eigenfunction sign flip and order switch problems of FAISD. We also avoid the costly space search of MT through all possible sample vertex triplets that define Möbius transformations for genus zero surfaces. Our algorithm can work with any genus.

On a 2.3GHz i5 laptop, it takes 7 seconds to obtain our Stage 4 map and then 23 more seconds to compute the full map over 12K vertices of a SCAPE model. The corresponding numbers for a 4K TOSCA Wolf are 2 seconds and 1 sec, whereas 27K



Figure 7. Extracted symmetry axis curves on TOSCA (first two rows) and SCAPE (last row) models along with a subset of our matches for which ground-truth correspondence is known.

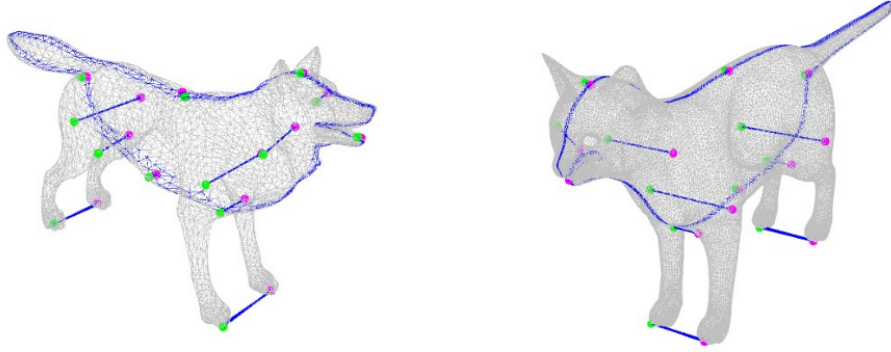


Figure 8. Our result set consisting of symmetry axis curve and symmetric correspondences on TOSCA Wolf and Cat models. Symmetric correspondences are restricted to vertices whose ground-truth matches are known.

Table 2. Correspondence rates (first 3 column from the left) and mesh rates (last 3 columns from the left) for SCAPE and human-shaped classes of TOSCA for MT, FAISD, and our proposed method.

	MT	FAISD	Ours	MT	FAISD	Ours
Victoria	83%	96%	78%	63%	100%	75%
Gorilla	-	100%	98%	-	100%	100%
David	82%	96%	95%	57%	100%	100%
Wolf	100%	100%	97%	100%	100%	100%
Cat	66%	96%	65 %	54%	100%	64%
Michael	87%	97%	88%	75%	100%	90%
SCAPE	82%	97%	81%	72%	98%	75%

Cat requires 27 and 105 seconds. This nearly-quadratic growth in time makes our algorithm practical to map high resolution models to themselves. Note that symmetry axis computation happens instantly as we traverse all vertices in linear time while checking the hybrid labeling situation in their 1-ring neighborhood. Completing Stage 4 is sufficient in order to perform this axis computation operation, e.g., 7 seconds for the symmetry axis curve of a SCAPE model.

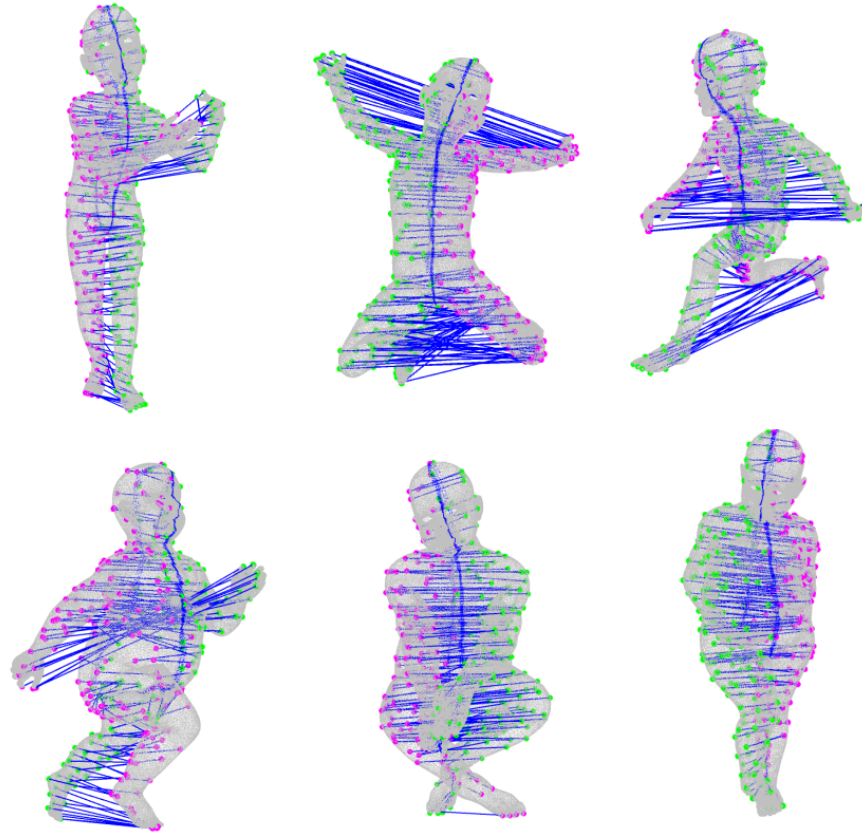


Figure 9. Our Stage 4 mapping along with the symmetry axis curve for thin kid (first row) and fat kid (second row) models from the KIDS dataset.

5. Limitations

Although our method is sufficiently generic handling popular human and four-legged animals, it will fail if the input model does not possess bilateral symmetry with four extreme points, e.g., octopus model. Although we did not encounter a case in practice where the extreme points are wrongly detected by taking the first 4 samples of the farthest point sampling procedure [Eldar et al. 1997], i.e., tip of hands and feet is not selected, it is theoretically possible to start with the wrong sample set, an error that cannot be recovered later by the upcoming stages. We finally note our global symmetry method will fail under partial symmetry cases where self-similarities occur only on parts of a shape.

6. Conclusion and Future Work

We presented a fast and robust algorithm for the global reflective intrinsic symmetry problem. By proper selection of base vertices and the corresponding feature vectors in coarse-to-fine manner, we were able to produce promising symmetric maps at sparse and dense levels. As a by-product, we instantly computed the symmetry axis curve that is invariant to isometric deformations. Our method requires no initial landmark matches, is insensitive to surface triangulation, and can be applied to shapes with arbitrary genus and resolution. Our performance is on a par with the more advanced and slower state-of-the-art methods we compared with.

One future work direction is to enhance the region selection mechanism in order to consider partial and rotational symmetries. While constructing k -nearest neighbor graph is mostly sufficient for the extraction of geodesic distances on point clouds, one can investigate alternative and more robust solutions to extract symmetry on this input type. Another future work can address data compression issue since symmetry can be interpreted as information redundancy. Finally, detection of symmetric partner points can help in the design of a symmetry-aware sampling algorithm that can be useful for various geometry processing tasks such as pairwise shape correspondence.

Acknowledgements

This project has been supported by TUBITAK under the project EEEAG-119E572.

References

- ANGUELOV, D., SRINIVASAN, P., KOLLER, D., THRUN, S., RODGERS, J., AND DAVIS, J. 2005. Scape: shape completion and animation of people. In *ACM SIGGRAPH 2005 Papers*. 408–416. [9](#)
- BRONSTEIN, A. M., BRONSTEIN, M. M., AND KIMMEL, R. 2008. *Numerical geometry of non-rigid shapes*. Springer Science & Business Media. [9](#)
- DE SILVA, V., AND TENENBAUM, J. B. 2002. Global versus local methods in nonlinear dimensionality reduction. In *NIPS*, vol. 15, 705–712. [3](#)
- DESSEIN, A., SMITH, W. A., WILSON, R. C., AND HANCOCK, E. R. 2017. Symmetry-aware mesh segmentation into uniform overlapping patches. In *Computer Graphics Forum*, vol. 36, 95–107. [1](#)
- ELDAR, Y., LINDENBAUM, M., PORAT, M., AND ZEEVI, Y. Y. 1997. The farthest point strategy for progressive image sampling. *IEEE Transactions on Image Processing* 6, 9, 1305–1315. [3](#), [8](#), [15](#)
- GRAUS, J., JACOBSON, A., AND GINGOLD, Y. 2021. Interacting with self-similarity. *Computer-Aided Design* 130, 102931. [3](#)
- JIANG, W., XU, K., CHENG, Z., AND ZHANG, H. 2013. Skeleton-based intrinsic symmetry detection on point clouds. *Graphical Models* 75, 4, 177–188. [3](#)

- KIM, V. G., LIPMAN, Y., CHEN, X., AND FUNKHOUSER, T. 2010. Möbius transformations for global intrinsic symmetry analysis. In *Computer Graphics Forum*, vol. 29, Wiley Online Library, 1689–1700. [2](#), [3](#), [4](#), [9](#), [11](#)
- LIU, T., KIM, V. G., AND FUNKHOUSER, T. 2012. Finding surface correspondences using symmetry axis curves. In *Computer Graphics Forum*, vol. 31, Wiley Online Library, 1607–1616. [1](#)
- LIU, X., LI, S., LIU, R., WANG, J., WANG, H., AND CAO, J. 2015. Properly constrained orthonormal functional maps for intrinsic symmetries. *Computers & Graphics* 46, 198–208. [2](#)
- MITRA, N. J., GUIBAS, L. J., AND PAULY, M. 2007. Symmetrization. *ACM Transactions on Graphics (TOG)* 26, 3, 63–es. [3](#)
- MITRA, N. J., PAULY, M., WAND, M., AND CEYLAN, D. 2013. Symmetry in 3d geometry: Extraction and applications. In *Computer Graphics Forum*, vol. 32, Wiley Online Library, 1–23. [2](#)
- NAGAR, R., AND RAMAN, S. 2018. Fast and accurate intrinsic symmetry detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 417–434. [2](#), [9](#), [11](#)
- OVSJANIKOV, M., SUN, J., AND GUIBAS, L. 2008. Global intrinsic symmetries of shapes. In *Computer graphics forum*, vol. 27, 1341–1348. [2](#)
- RAVIV, D., BRONSTEIN, A. M., BRONSTEIN, M. M., AND KIMMEL, R. 2007. Symmetries of non-rigid shapes. In *2007 IEEE 11th International Conference on Computer Vision*, IEEE, 1–7. [2](#)
- RAVIV, D., BRONSTEIN, A. M., BRONSTEIN, M. M., AND KIMMEL, R. 2010. Full and partial symmetries of non-rigid shapes. *International journal of computer vision* 89, 1, 18–39. [2](#)
- RAVIV, D., BRONSTEIN, A. M., BRONSTEIN, M. M., KIMMEL, R., AND SAPIRO, G. 2010. Diffusion symmetries of non-rigid shapes. In *Proc. 3DPVT*, vol. 2. [2](#)
- REN, J., MELZI, S., OVSJANIKOV, M., AND WONKA, P. 2020. Maptree: recovering multiple solutions in the space of maps. *ACM Trans. on Graph.* 39, 6, 264–1. [2](#)
- RODOLA, E., ROTA BULO, S., WINDHEUSER, T., VESTNER, M., AND CREMERS, D. 2014. Dense non-rigid shape correspondence using random forests. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4177–4184. [9](#)
- SAHILLIOĞLU, Y., AND KAVAN, L. 2016. Detail-preserving mesh unfolding for nonrigid shape retrieval. *ACM Trans. on Graph.* 35, 3, 1–11. [3](#)
- SAHILLIOĞLU, Y., AND YEMEZ, Y. 2012. Minimum-distortion isometric shape correspondence using em algorithm. *IEEE transactions on pattern analysis and machine intelligence* 34, 11, 2203–2215. [3](#)
- SAHILLIOĞLU, Y. 2020. Recent advances in shape correspondence. *The Visual Computer* 36, 8, 1705–1721. [1](#)

- SHI, Y., HUANG, J., ZHANG, H., XU, X., RUSINKIEWICZ, S., AND XU, K. 2020. Symmetry-net: learning to predict reflectional and rotational symmetries of 3d shapes from single-view rgb-d images. *ACM Transactions on Graphics (TOG)* 39, 6, 1–14. 3
- WANG, H., AND HUANG, H. 2017. Group representation of global intrinsic symmetries. In *Computer Graphics Forum*, vol. 36, 51–61. 2
- WANG, W., MA, J., XU, P., AND CHU, Y. 2019. Intrinsic symmetry detection on 3d models with skeleton-guided combination of extrinsic symmetries. In *Computer Graphics Forum*, vol. 38, 617–628. 3
- XU, K., ZHANG, H., TAGLIASACCHI, A., LIU, L., LI, G., MENG, M., AND XIONG, Y. 2009. Partial intrinsic reflectional symmetry of 3d shapes. In *ACM SIGGRAPH Asia 2009 papers*. 1–10. 2
- XU, K., ZHANG, H., JIANG, W., DYER, R., CHENG, Z., LIU, L., AND CHEN, B. 2012. Multi-scale partial intrinsic symmetry detection. *ACM Trans. on Graph.* 31, 6, 181:1–181:10. 2
- XU, H., HUI, K. H., FU, C.-W., AND ZHANG, H. 2019. Computational lego technic design. *ACM Trans. on Graph.* 38, 6. Article no. 196. 1
- YOSHIYASU, Y., YOSHIDA, E., AND GUIBAS, L. 2016. Symmetry aware embedding for shape correspondence. *Computers & Graphics* 60, 9–22. 1

Index of Supplemental Materials

We share the source code and executables of our paper, and the outputs, at: <http://ceng.metu.edu.tr/~ys/pubs/sym.zip>.

Author Contact Information

Simge Tekin
Middle East Technical University
Computer Engineering Dept.
Ankara, Turkey 06800
tekin.simge@metu.edu.tr

Yusuf Sahillioğlu
Middle East Technical University
Computer Engineering Dept.
Ankara, Turkey 06800
ys@ceng.metu.edu.tr
<http://www.ceng.metu.edu.tr/~ys>

Simge Tekin and Yusuf Sahillioğlu, A Method for Extracting Symmetry from Meshes, *Journal of Computer Graphics Techniques (JCGT)*, vol. vol, no. issue, 1–1, year
<http://jcgt.org/published/vol/issue/num/>

Received: submitted
Recommended: accepted
Published: published

Corresponding Editor: Editor Name
Editor-in-Chief: Marc Olano

© year Simge Tekin and Yusuf Sahillioğlu (the Authors).

The Authors provide this document (the Work) under the Creative Commons CC BY-ND 3.0 license available online at <http://creativecommons.org/licenses/by-nd/3.0/>. The Authors further grant permission for reuse of images and text from the first page of the Work, provided that the reuse is for the purpose of promoting and/or summarizing the Work in scholarly venues and that any reuse is accompanied by a scientific citation to the Work.

